System Modelling and Simulation
S. Tzafestas, A. Eisinberg, L. Carotenuto (Editors)
Elsevier Science Publishers B.V. (North-Holland)
© IMACS, 1989

GRAMAN: A Graphic System for Manufacturing System Design

J.L. Villarroel, J. Martínez, M. Silva

Dpto. de Ingeniería Eléctrica e Informática, Universidad de Zaragoza
María Zambrano, 50, 50015 ZARAGOZA (SPAIN)

1. INTRODUCTION

GRAMAN is a graphic system for describing manufacturing systems. It is integrated into a wider environment for CAD of computer systems which evaluates performance and controls manufacturing systems, whose structure is shown in figure 1.

A manufacturing system is described in GRAMAN by distinguishing between the *plant description* with its elements (machines, transport and handlers, stores, etc.) and the structural relation between them (basically restrictions imposed by the layout), and the *description of the work plans* (definition of sequences of orders about manufacturing and transport; materials and production resources involved; etc.).

Generally, modifications to a manufacturing plant are rare, whereas work plans can be in almost constant evolution. The decision to separate the manufacturing plant description from that of the work plans means that any modifications will affect only one part of the design. Furthermore, in our opinion, this approach is more intuitive and easier to understand than a design integrating the two areas.

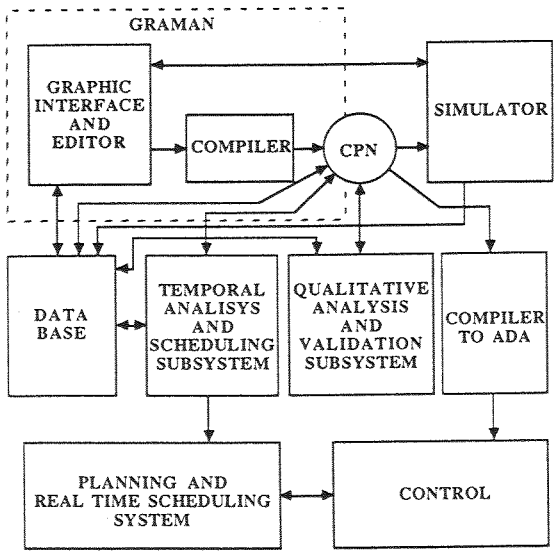

Figure 1: Computer aided design system for concurrent systems integrating GRAMAN.

GRAMAN generates an internal model for describing manufacturing systems which consists of a *colored Petri net* (CPN, in the sequel). This description acts as an input to the rest of the environment tools: simulator, quantitative and qualitative analyzers, and code generator. For reasons of space, and conciseness, this paper will restrict itself to describing the GRAMAN system. A presentation of CPN's and of their advantages for manufacturing system design can be found in [1].

From the user's point of view, GRAMAN describes a manufacturing plant by defining a set of *building blocks* and their *interconnections*. A building block can be associated with a production resource of the system (machine, manipulator, store, etc.) or can refer to a subsystem. In the latter case, the corresponding block must be refined by using the same descriptive elements (interconnected building blocks). Therefore, this is a top-down description of the plant.

Production plans are described very simply by using Petri nets. This type of model facilitates the description of sequences of orders, alternative paths, assembly or disassembly instruction, etc. and allows design by stepwise refinements. GRAMAN compiles and links the descriptions of the plant and the work plans to generate a single model (CPN).

This system will be presented as follows. The ways of describing a production plant and work plans are presented in §2 and §3, respectively. The generation of models which serve as inputs to the other environment tools is discussed in §4. In particular, the generation of models for the simulator and code generator is discussed. Finally, in §5, our system is placed in context with other work done with similar objectives.

2. DESCRIPTION OF THE PRODUCTION PLANT

The description of a production plant can be made directly by using CPN. However, to facilitate the job of the designer we have defined an aplication oriented high level graphic language.

In GRAMAN, the description of a production plant consists of the description of the production resources it has, and the structural relations between them. Two elements are said to be structurally related if there can exist

a flow of dynamic elements (tools, dies, moulds, transport vehicles, pallets, materials or orders) between them.

Each production system resource is represented by a building block inside which the associated activity(ies) is(are) specified. These describe the resource functionally. Figure 2 shows the icons corresponding to the building blocks, the subicons associated with activities and the attributes associated with each type of resource. All the activities have the attributes **order/no_order** except for transport (always **no_order**). This attribute refers to the possibility of a resource obeying explicit orders, or simply following the flow of material.

Input and output of material to and from a resource are described by what we will call *interchange points*, which are graphically represented as circles situated on the perimeter of the corresponding icon. The interchange points define a sequential input or output of material, which corresponds to an input or output (physical or logical) of the physical resource.

The structural relations define the possible paths for flows of material and dynamic resources in the system. The GRAMAN system, as it is currently defined, only handles material flows (e.g. tools, dies, moulds and pallets are not considered). Structural relations are represented by oriented arcs whose origin and destination are interchange points of different building blocks. All arcs which meet at a given interchange point must have the same orientation (in or out), thus an interchange point is defined as input or output.

The plant of a manufacturing system can be described *top-down*, by including building blocks not specified at the level being considered (*abstract building blocks*). The internal description (refinement) of an abstract block is done by interconnecting building blocks, which can in turn be abstract. The *interchange points* are the interface of an abstract block with its surroundings, and must be respected in the refinement process. Any connection between internal and external building blocks and an abstract block must be via the abstract block's interchange points.

Resource type	Icon	Activity	Subicon	Attributes	
Passive resource (stores, buffers)	○			Capacity	Identifier
Active resources	□	Manufacture		Order/ no_order	
		Disassemble			
		Assemble		Parts num. Input policy	
		Quality control		Bad material output	
		Transfer			
		Transport		Capacity	

Figure 2: Building blocks, activities and associated attributes.

Figure 3 shows the description of an assembly and manufacturing cell. It consists of two stores, one input (IS) and one output (OS), a drilling machine (M1), an assembler (M2) which operates on two parts and whose loading policy is sequential, loading the first part through input "a" and the second through input "b", a lathe (M3) with an input buffer and an output buffer (B1 and B2) and a robot (R_p) which loads and unloads the machines. This example will be resumed in §4 and §5, where the cell design will be completed.

Automatic guided vehicles (AGV) receive special treatment in GRAMAN, given their specific characteristics. A system of this type consists of:

- A static resource: transport network.
- Various dynamic resources: automatic guided vehicles moving through the network.

A system of this type does not lend itself to description by a single building block because of its complexity and its variable behavior, which depends on the network design. Therefore, GRAMAN contains a different design methodology for AGV networks. To define an AGV transport system, it is necessary to describe:

i) The topology of the transport network, which can be compared to a theoretical building block representing the resource, and

ii) A set of attributes: network dimension, number and speed of vehicles and their initial position.

To describe a network four elements will be used: sections (of line), control points, nodes and interchange

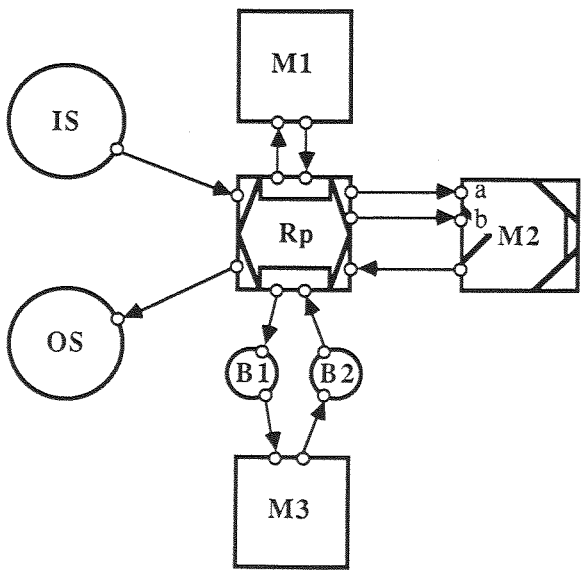

Figure 3: Manufacturing and assembly cell. A robot feeds three machines.

points. The *sections* will be graphically represented by straight line segments, connected by control points, interchange points and nodes. The *nodes* are elements representing curves, junctions or bifurcations of the network. The topology of a circuit can be represented by the sections and nodes. The control points are the elements which divide a circuit into segments which, in order to avoid collisions, can contain a maximum of one vehicle at a time. By default, each end of a node which represents a junction or bifurcation is associated with a control point. Therefore, only additional control points need to be defined. Figure 4 gives the graphic representation of these elements, together with an example of the design of an AGV network. The AGV model is connected to the rest of the design in a natural way via its interchange points.

Representing static resources by building blocks and AGV systems by a topological description of the transport network brings the production plant design close to the actual plant layout. This makes it much easier to interpret the final design.

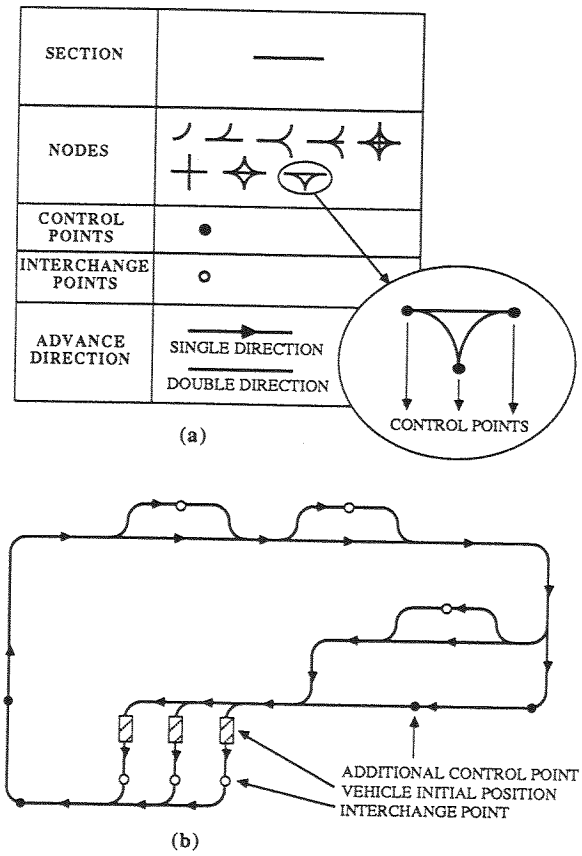

Figure 4: AGV design:
(a) Design elements
(b) Design example.

3. PRODUCTS AND MANUFACTURING

Production follows *work plans*, each of which has the goal of manufacturing one product. A work plan describes the operations to be performed and the resources and materials required.

A *material* can be characterized by two attributes which specify its *identity* and its *production status*. The plant receives a number of orders about manufacturing, which we will call *primary orders*. Obeying a primary order involves carrying out the corresponding work plan, defined as an attribute of the order. Executing a *work plan* for a given order generates a series of *secondary orders* about performing operations in the system resources. A secondary order has the associated attributes *operation type* and *list of materials* required to perform it.

In GRAMAN, work plans are defined by Petri nets. In this context, the elements of a Petri net have the following semantics:

- *Places*. Represent partial-distributed states in the execution of the work plan.
- *Transitions*. Represent the execution of operations and, therefore, a possible evolution of the work plan's state.

To permit the hierarchical design of a work plan, *abstract transitions* called *macrotransitions* are used. These represent partial work plans which, in turn, must be described (refined) by Petri subnets.

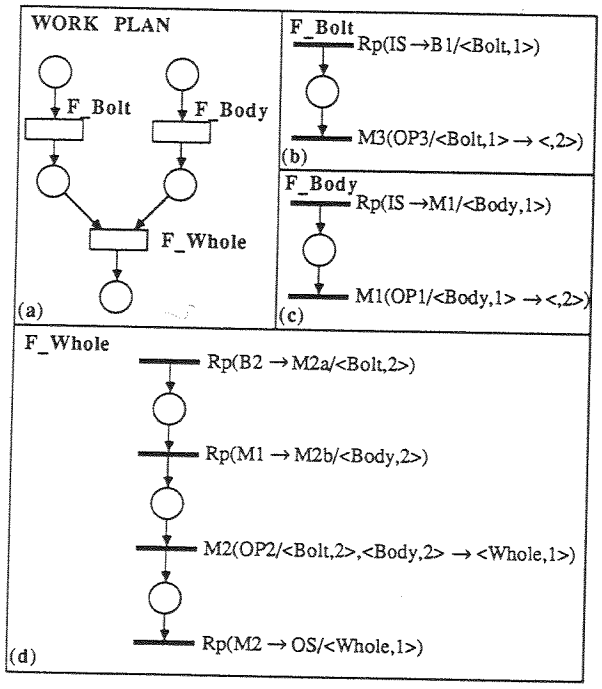

Figure 5: Work plan example for the production plant from figue 3.

A transition is labeled with the description of the operation it represents. This specifies firstly the resource where the operation is to be performed (receiver of the secondary order which provokes the execution) and, secondly, in brackets, the identification of the operation (origin and destination resource identifiers, in transfer operations) and the list of materials required and of the transformations they undergo. A macrotransition is labeled with an identifier which refers to the subnet which constitutes its refinement.

Figure 5 shows the definition of a simple work plan for the cell described in figure 3. The work plan, figure 5a, is described at a first level of abstraction by macrotransitions. The "macro_operations" F_Bolt and F_Body must have concluded before F_Whole is performed. F_Bolt and F_Body can be executed concurrently. Figure 5b and 5c show the refinement of macrotransitions F_Bolt and F_Body, respectively. Note that in the description of the manufacturing operations for M1 and M3, the change of state which the parts undergo is specified. Finally, figure 5d shows the refinement of the macrotransition F_Whole. It is a subnet which describes a sequence of operations: loading M2 through entry "a" with a part "Bolt" (<Bolt,2>), followed by loading a "Body" (<Body,2>) through entry "b"; assembly of the two parts to give <Whole,1>; unloading of M2 by robot R_p, which transports the assembled product to store "OS".

4. GENERATING AN INTERNAL MODEL

In GRAMAN the production plant and the work plans are described separately. However, they are interdependent since a work plan contains orders which must be executed by the resources in the plant and, furthermore, the evolution of a work plan depends on the end of execution in the plant of the operations associated with these orders.

The final aim of GRAMAN is to construct models of manufacturing systems which will serve as inputs for the rest of the tools in the CAD environment outlined in figure 1. These models will differ slightly depending on the tool in question. Thus, for example, a model for the performance evaluator simulator must contain an estimation of the time taken for each operation in the work plan, whereas a model for the code generator must foresee details of the interface with the controllers of elements in the plant. Because of the lack of space, his paper restricts itself to briefly presenting the characteristics which are common to these models, and the basic concepts of describing production plants (§2) and work plans (§3).

GRAMAN compiles a production plant description (figure 3) by generating a subCPN. A predefined subCPN, parametrized by the structural relations which affect the block, is assigned to each block. Figure 6 shows the subCPN's associated with the building blocks which appear in figure 3. These subnets represent possible states of each resource (e.g. free, waiting to be loaded, operating, waiting to be unloaded, etc.), the possible evolutions between these states (transitions of the subnet) and the restrictions on the occurrence of each of these evolutions (preconditions of each transition). These subnets are connected according to the structural relations (arcs) described in the plant model. This is done by fusing the transitions representing actions to be synchronized, which correspond to structurally connected pairs of resources. To illustrate this idea, figure 6g shows the result of representing the structural relations defined between the robot R_p and assembly machine M2. The resulting subnet contains the fusion of the transitions which imply synchronization between the two resources during loading and unloading operations.

The models of the work plans are Petri nets. They must be connected to the overall plant model. Conceptually, this is done by "rendez-vous" type mechanisms. Firing a transition of a net modeling a work plan carries with it an associated production order. That order will start to execute in the plant when the required conditions are met (availability of necessary resources, materials, etc.). The model of the work plan will not finish firing the transition until the operation triggered by the corresponding order has finished. The "rendez-vous" occurs, therefore, when an order is launched in a work plan (on firing a given transition) and when the plant model meets the conditions necessary for the order to be executed.

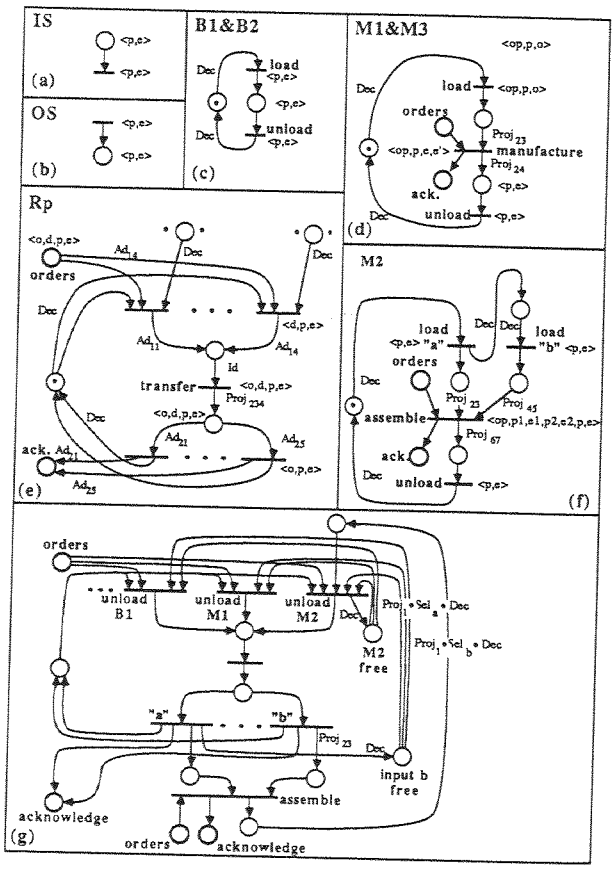

Figure 6: Sub-CPNs which model the building blocks of the example from figure 3 and coordinaton model generated by linking robot Rp and machine M2.

5. OTHER SYSTEMS

This section contains a brief comparative study of the GRAMAN design methodology and those described in [2] and [3].

The "Planning System MOSYS of IPK in Berlin" [2] divides the design process into three different phases: (1) functional description of the production process, (2) assignment of resources, and (3) topological description of the manufacturing plant. The first phase describes the production process by means of interconnected functional blocks. In the second phase, each functional block is assigned the resource which performs the activity described. The topological description contains the physical location of resources and the description of the AGV transport network.

[3] presents a design methodology based on colored Petri nets, developed in the CASPAIM project in the "Laboratoire d'Informatique Industrielle de l'I.D.N.". The production system is first described by a colored Petri net (Pregraph), which is the result of combining a set of predefined basic structures describing the system's possible activities (manufacture, transfer, assembly, ...). Each of these structures has an associated subnet at a lower level of abstraction which describes the activity represented by the structure. From the pregraph and the internal models of the basic structures, a coordination and simulation model is generated, which must be modified interactively to include the restrictions of the production plant (mutually exclusive or synchronized activities).

To situate our approach in relation to the two above, we will focus on three basic aspects:

i) *Generated model.* Both GRAMAN and the CASPAIM methodologies generate coordination and simulation models based on colored Petri nets. MOSYS, on the other hand, generates a simulation model, and another performance model based on queueing networks, but does not generate a coordination model. Both GRAMAN and CASPAIM generate coordination models which allow both qualitative and quantitative analysis of the design. Since there are tools for automatic code generation from the nets, validation of the coordination model also implies a validation of the structure of the control software.

ii) *Design methodology.* The CASPAIM and MOSYS methodologies describe the activities in the work plans first (the former with nets, the latter with functional blocks), and later establish the restrictions imposed by the system's physical resources. MOSYS goes further by establishing a topological model of the plant to describe the AGV transport systems. GRAMAN permits the separate design of work plans and plant, by making explicit the production processes (hidden in CASPAIM as they are all integrated into a single model) and the structural relations between resources (hidden in MOSYS and CASPAIM). The structural relations defined in GRAMAN describe cooperation between related resources (loading one must synchronize with unloading the other) as well as possible paths of material flow. The work plans define, though not exhaustively, the route followed by materials in the plant. The redundancy existing between the work plans and the production plant description permits the work plans to be verified, or reveal whether they fit the plant.

iii) *Production orders.* GRAMAN is the only one of the three to include this concept. From our point of view, they are important elements in generating the control software and in maintaining the design, for several reasons:

- They define a simple means of communication between the control and coordination system and a production planning system.
- They allow the status of production to be easily monitored (status of execution of primary orders).
- They make it easy to modify or to include new work plans since they define a clear interface with the design.

7. CONCLUSIONS

The GRAMAN system is designed to be integrated into a broader CAD environment for concurrent systems, which uses Petri nets as the basic model. This environment includes analysis, validation and automatic implementation generation tools [4].

The methodology developed for GRAMAN permits separate design of the production plant by means of *interconnected building blocks* and of the work plans by means of *Petri nets*. Apart from basic operations, GRAMAN considers assembly and disassembly operations, and allows the design of specific transport systems, such as AGV transport networks.

This separation is maintained in the generated model, by means of two subCPN's joined by "rendez-vous" type mechanisms. One of these subnets, the plant model, remains unchanged by modifications or extensions of the *work plans*. The other subnet, generated from the work plans, manages the sending of *production orders* to the plant.

This paper does not deal with decision problems arising in the models generated by GRAMAN (e.g. choice between alternative work plans; assignment of a resource between various pending orders; etc.). Our Department is currently working on the design of decision systems using artificial intelligence techniques [5]. This work, which will allow the "Planning and Real Time Scheduling System" of the environment in figure 1 to be redesigned, complement that presented here.

GRAMAN consists of a specific graphic editor and a compiler which translates the design into a colored Petri net. This is the input for the simulator, implementation generator, and the analysis and validation tools. In the future we foresee a graphic simulation animator.

GRAMAN is currently being implemented on a UNIX workstation, using the standards GKS and X-windows.

ACKNOWLEDGEMENTS

This work was carried out within projects IT-2/86 of CONAI of the Diputación General de Aragón and PA 86-0028 of the Comisión Interministerial de Ciencia y Tecnología (CICYT).

REFERENCIAS

[1] F. MARTINEZ, H. ALLA & M. SILVA: "Petri Nets for the Specification of FMSs". Modelling and Design of Flexible Manufacturing Systems. Edited by A. Kusiak, ELSEVIER, Amsterdam 1986, pp. 389-406.

[2] G. SELIGER, B. WIENEKE-TOUTAOUI & M. RABE: "Integrated Modelling of Manufacturing Systems for Planning Purposes". The Winter Annual Meeting of the American Society of Mechanical Engineers. Boston 1987, pp. 405-416.

[3] J.P. BOUREY: "Structuration de la Partie Procedurale du Système de Commande de Cellules de Production Flexibles dans l'Industrie Manufacturiere". These docteur d'Université, Université des Sciences et Techniques de Lille Flandres Artois. Lille 1988.

[4] M. SILVA & J. MARTINEZ: "A Software Environement for Systems Design with High Level Petri Nets and their Implementations". Int. Seminar on Applicability of Petri Nets to Operation Research, Milano 1986, pp. 95-127.

[5] J. MARTINEZ, P.R. MURO, M. SILVA, S.F. SMITH & J.L. VILLARROEL: "Merging Artificial Intelligence Techniques and Petri Nets for Real Time Scheduling and Control of Production Systems". To appear in Proc. of 12th IMACS World Congress on Scientific Computation. Paris 1988.